The discord bot Mee6 contains a leveling system feature that users of a discord server can participate in. Experience is rewarded to members at most once per minute for sending a message in the server. The amount of experience gained for a single message can range from 15 to 25, inclusive. This can be described as the equivalent statement 15 + $\lfloor X \rfloor$ where X is a random variable distributed uniformly. That is,

$$X \sim U(0,11)$$

To create a more dynamic (read: pay to win) reward system, four key areas have been identified where experience rewards can be modified. They are as follows:

- Base experience gain (15) [b]
- Random bonus maximum (10) [r]
- Cooldown between messages that qualify for experience rewards (60 seconds)
- Random bonus probability distribution, “luck” (g(X)) [U]

The objective for all four parameters is to have a buff modifier to the parameter that is initially zero, ideally provides roughly linear improvements per point of the buff, and can be set to any non-negative value less than 1 million (which would achieve the top level in a mere six minutes) without causing problems, such as a divide by zero error would.

The first two features can easily be given linear modifiers (15 + n, 10 + m), and the cooldown between messages can be given a linear buff per cooldown point using the formula below, explained further at the source (LoL: Ability Haste):

$$\text{Reduced cooldown} = \text{Base Cooldown} \times \frac{100}{100 + \text{Haste}}$$

Following a similar formula for the “luck” modifier for the otherwise uniform distribution of the random variable. Using the following transformation:

$$U = g(X) = X^a \qquad a = \frac{100}{100 + luck} \qquad 0 \le luck \le 1{,}000{,}000 \qquad X \sim U(0,1)$$

X is now distributed uniformly from 0 to 1. Multiplication by the random bonus maximum is evaluated after the calculation.

Using the CDF technique knowing that $f_x(x) = \begin{cases} 1, & 0 < x < 1 \\ 0, & otherwise \end{cases}$

$$F_U(u) = P(U \le u) = P(X^a \le u) = P\left(X \le u^{\frac{1}{a}}\right) = F_X\left(u^{\frac{1}{a}}\right)$$

$$f_U(u) = \frac{d}{du}\big(F_U(u)\big) = f_X\left(u^{\frac{1}{a}}\right) * \frac{d}{du}\left(u^{\frac{1}{a}}\right) = 1 * \frac{1}{a} * u^{\left(\frac{1}{a}-1\right)}$$

For this particular case, the useful information to have is the expected value of the random variable as a function of the user’s luck.

$$E[U] = \int_{u=0}^{u=1} u * f_U(u)du = \int_{u=0}^{u=1} \frac{u^{(1/a)}}{a} du = \frac{u^{\left(1+\frac{1}{a}\right)}}{\left(1+\frac{1}{a}\right)a}\Bigg|\begin{matrix} u = 1 \\ u = 0 \end{matrix} = \frac{1}{a+1}$$

Substituting in the value of a with the function of luck defined above,

$$E[U] = \frac{1}{a+1} = \frac{1}{\frac{100}{100 + luck} + 1} = \frac{1}{\frac{200 + luck}{100 + luck}} = \frac{100 + luck}{200 + luck} = 1 - \frac{100}{200 + luck}$$

With these calculations, the expected value of the random variable over the range of luck from 0 (no modifiers) to 1000 (maximum bonus from a single role) is as follows:

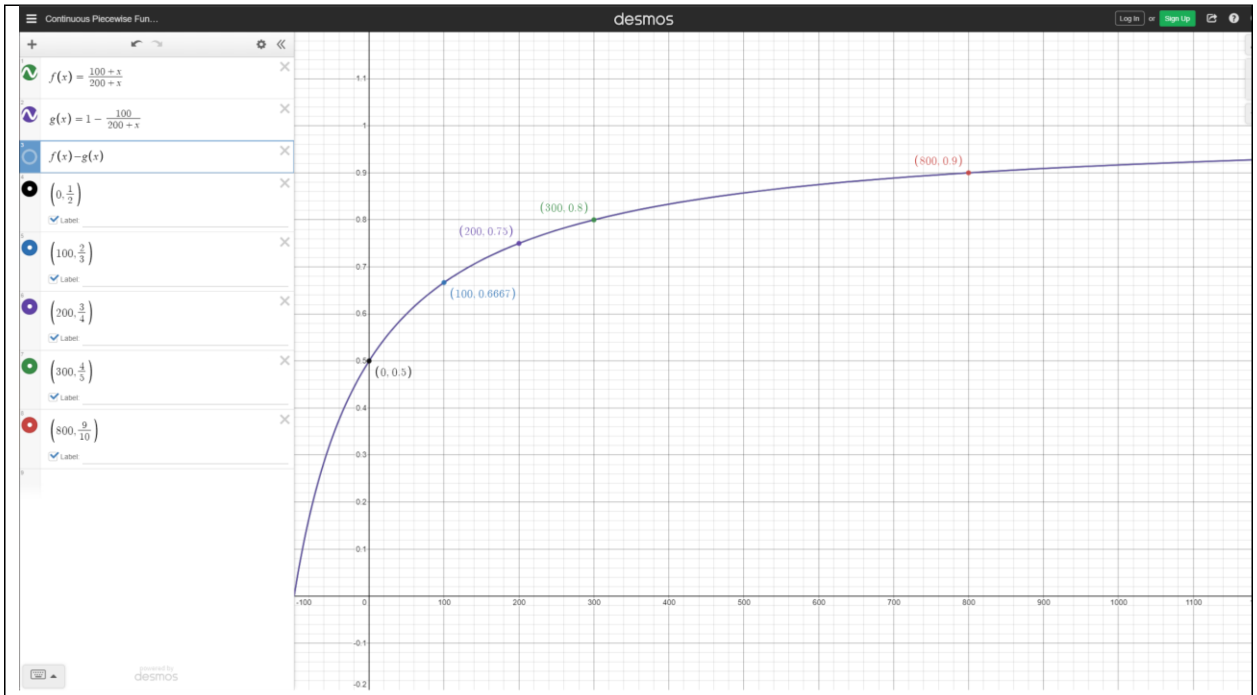


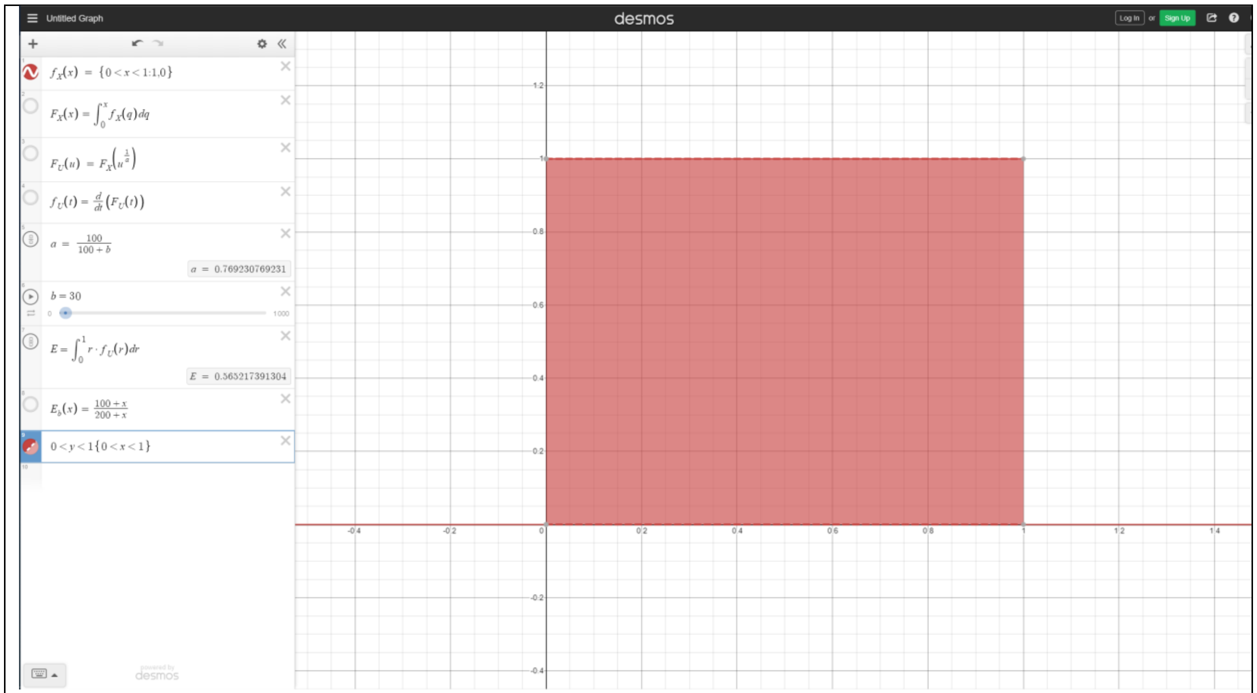
Fig. 1: the expected value of the random variable modified by the luck of a role.
https://www.desmos.com/calculator/jyeaiydkhg

The expected value acts as a long term projection of the effectiveness of the random bonus maximum and its modifier.

With the new distribution defined by U, the experience gained for any qualifying message, after modifications as applicable, becomes

$$EXP = b + r * U$$

Appendix A: Individual distribution graphs



$f_X(x) = \{0<x<1:1,0\}$

$F_X(x) = \int_0^x f_X(q)dq$

$F_U(u) = F_X\left(u^{\frac{1}{a}}\right)$

$f_U(t) = \frac{d}{dt}\left(F_U(t)\right)$

$a = \frac{100}{100+b}$

$a = 0.769230769231$

$b = 30$

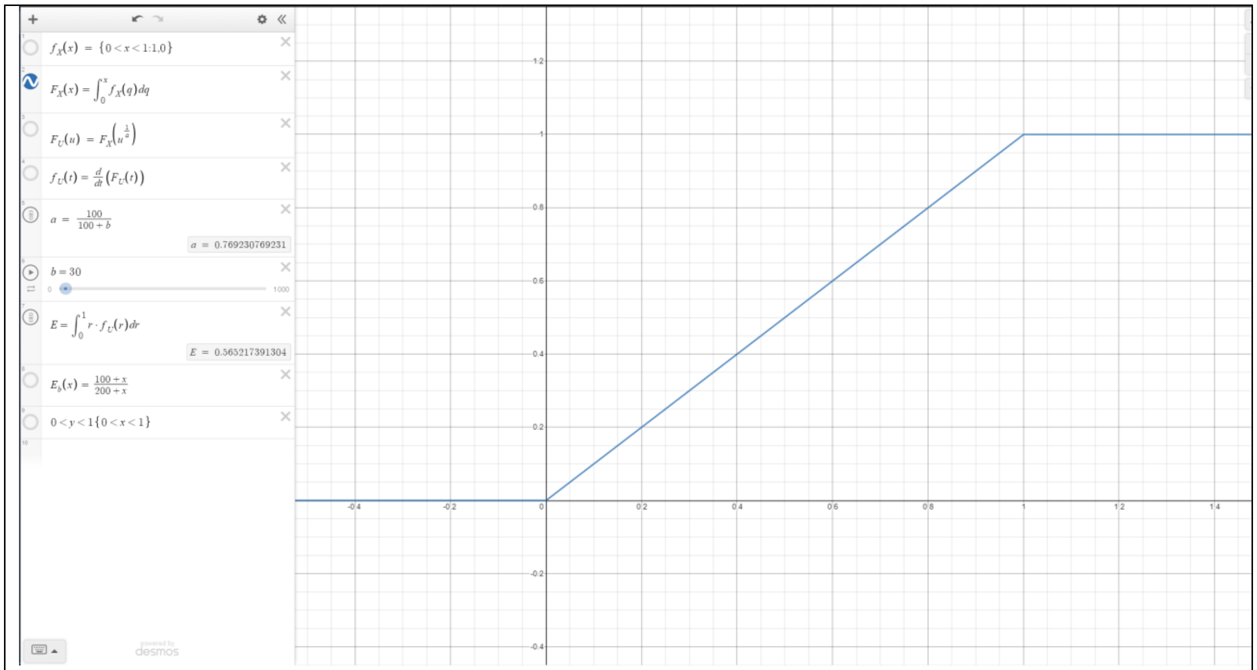
$E = \int_0^1 r \cdot f_U(r)dr$

$E = 0.565217391304$

$E_b(x) = \frac{100+x}{200+x}$

$0<y<1\{0<x<1\}$

Fig. 2: The probability density function of X~U(0,1)
https://www.desmos.com/calculator/ke18tlgjin



$f_X(x) = \{0<x<1:1,0\}$

$F_X(x) = \int_0^x f_X(q)dq$

$F_U(u) = F_X\left(u^{\frac{1}{a}}\right)$

$f_U(t) = \frac{d}{dt}\left(F_U(t)\right)$

$a = \frac{100}{100+b}$

$a = 0.769230769231$

$b = 30$

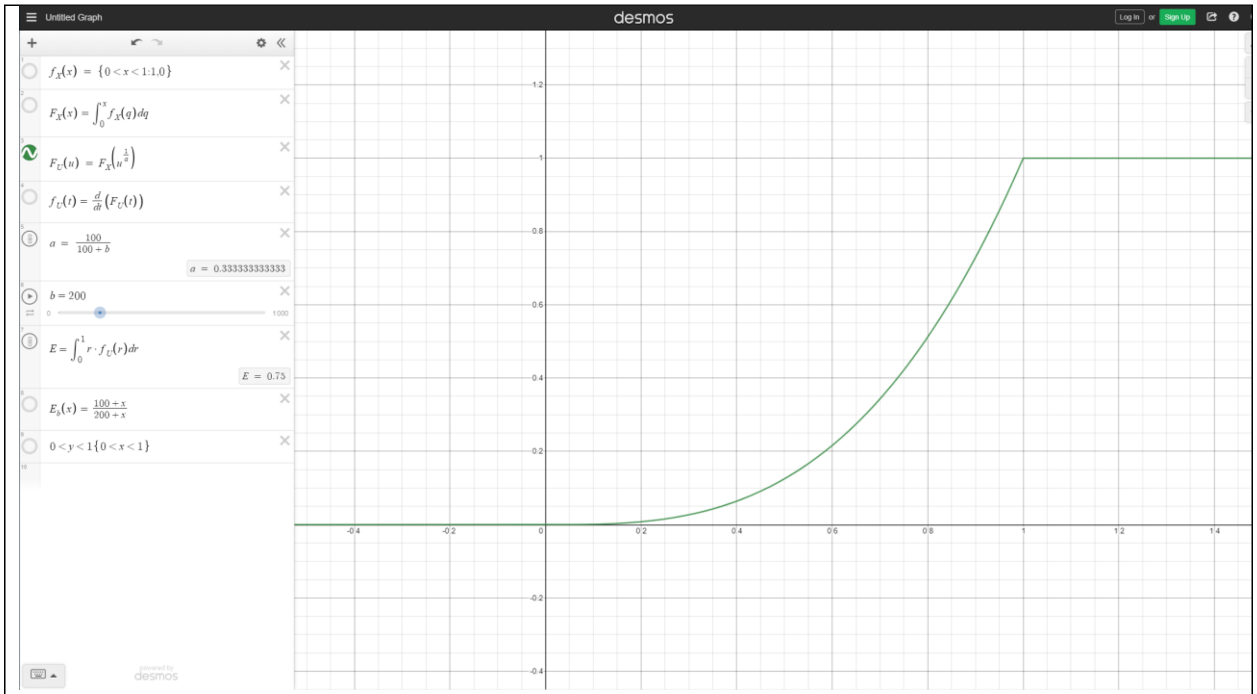
$E = \int_0^1 r \cdot f_U(r)dr$

$E = 0.565217391304$

$E_b(x) = \frac{100+x}{200+x}$

$0<y<1\{0<x<1\}$

Fig. 3: The Cumulative Distribution Function of X~U(0,1)
https://www.desmos.com/calculator/vtjb5rpoai



$f_X(x) = \{0 < x < 1 : 1, 0\}$

$F_X(x) = \int_0^x f_X(q)\,dq$

$F_U(u) = F_X\left(u^{\frac{1}{a}}\right)$

$f_U(t) = \frac{d}{dt}\left(F_U(t)\right)$

$a = \frac{100}{100+b}$

$a = 0.333333333333$

$b = 200$

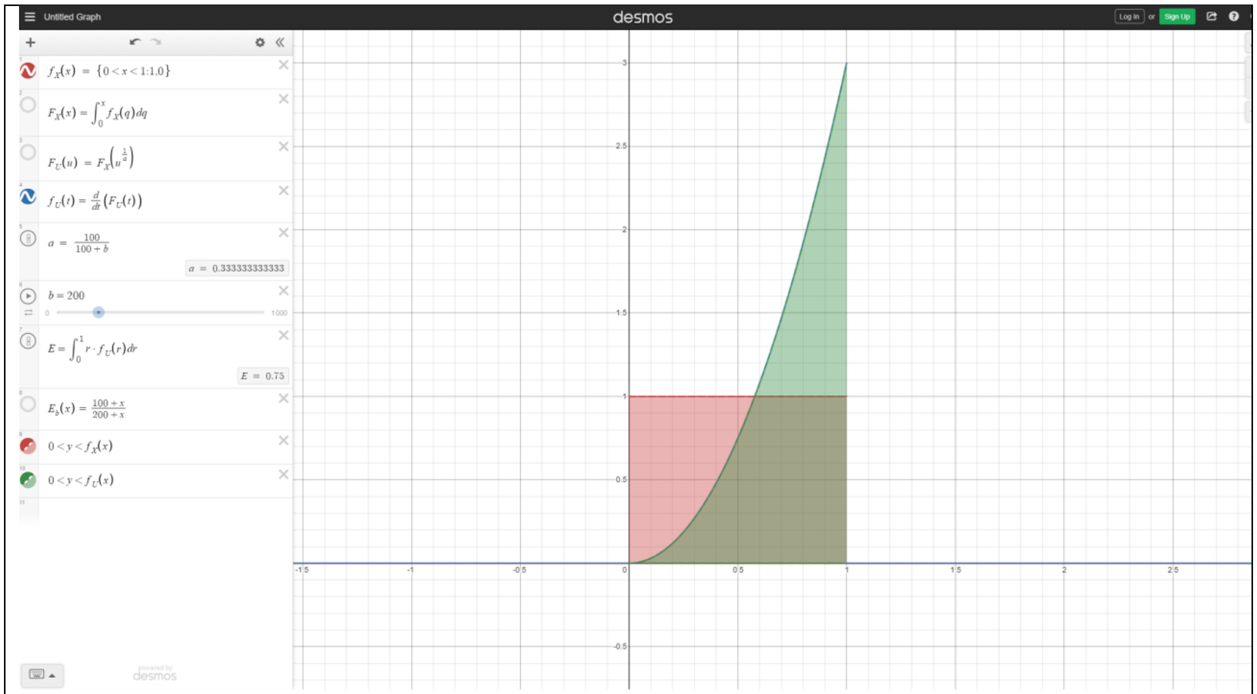
$E = \int_0^1 r \cdot f_U(r)\,dr$

$E = 0.75$

$E_b(x) = \frac{100+x}{200+x}$

$0 < y < 1\{0 < x < 1\}$

Fig. 4: The Cumulative Distribution Function of U = X^a for a luck value of 200 → a=1/3
https://www.desmos.com/calculator/ux3tjsajpx



$f_X(x) = \{0 < x < 1 : 1, 0\}$

$F_X(x) = \int_0^x f_X(q)\,dq$

$F_U(u) = F_X\left(u^{\frac{1}{a}}\right)$

$f_U(t) = \frac{d}{dt}\left(F_U(t)\right)$

$a = \frac{100}{100+b}$

$a = 0.333333333333$

$b = 200$

$E = \int_0^1 r \cdot f_U(r)\,dr$

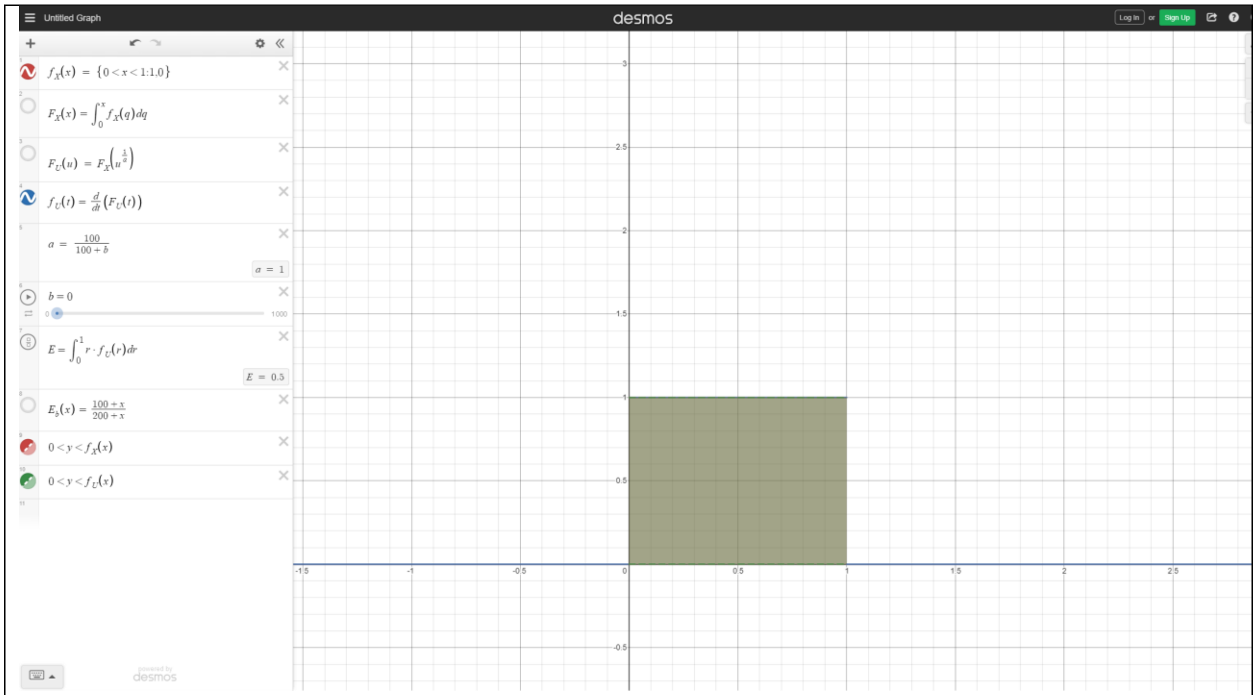
$E = 0.75$

$E_b(x) = \frac{100+x}{200+x}$

$0 < y < f_X(x)$

$0 < y < f_U(x)$

Fig. 5: The probability density functions for X (red, distributed uniformly) and U (green, the transformation of X). Notice that a significantly greater portion of the area under the curve appears to the left of 0.5 in the distribution of U.
https://www.desmos.com/calculator/3qaegg5nld



1 $f_X(x) = \{0 < x < 1 : 1, 0\}$

2 $F_X(x) = \int_0^x f_X(q)\,dq$

3 $F_U(u) = F_X\left(u^{\frac{1}{a}}\right)$

4 $f_U(t) = \frac{d}{dt}\left(F_U(t)\right)$

5 $a = \frac{100}{100 + b}$

$a = 1$

6 $b = 0$

0 1000

7 $E = \int_0^1 r \cdot f_U(r)\,dr$

$E = 0.5$

8 $E_b(x) = \frac{100 + x}{200 + x}$

9 $0 < y < f_X(x)$

10 $0 < y < f_U(x)$

Fig. 6: When the luck bonus is set to 0, the transformation makes no change to the uniform distribution, preserving the same experience for guilds that don’t chose to use this feature. https://www.desmos.com/calculator/zhsrmjdnff

Appendix B: Empirical confirmation code (node.js)

Program

```
// Beginning of program
"use strict";
const TRIAL_COUNT  = 10000;
function getExpectedValue(LUCK){
	let EV = 0;
	//returns a random value between 0 and 1
	function randResult(){
		return Math.pow(Math.random(), (100)/(100 + LUCK));
	}
	for(let i = 0; i < TRIAL_COUNT; i++){
		EV += randResult();
	}
	return EV / TRIAL_COUNT;
}
for(let luck = 0; luck < 1001; luck += 100){
	let EV = getExpectedValue(luck);
	console.log(`Expected value given luck =\t${luck}:\t${EV}\tIdeal:
${(100+luck)/(200+luck)}`);
}
// End of program
```

Execution output

```
Console output:
Expected value given luck =     0:      0.4966735510681596      Ideal: 0.5
Expected value given luck =     100:    0.6709645102270652      Ideal: 0.6666666666666666
Expected value given luck =     200:    0.7479804802867689      Ideal: 0.75
Expected value given luck =     300:    0.8014302848633157      Ideal: 0.8
Expected value given luck =     400:    0.8332899158503493      Ideal: 0.8333333333333334
Expected value given luck =     500:    0.8586513842725044      Ideal: 0.8571428571428571
Expected value given luck =     600:    0.8762311459099273      Ideal: 0.875
Expected value given luck =     700:    0.888499559802101       Ideal: 0.8888888888888888
Expected value given luck =     800:    0.9004658949275961      Ideal: 0.9
Expected value given luck =     900:    0.9091978481747085      Ideal: 0.9090909090909091
Expected value given luck =     1000:   0.9166335114831361      Ideal: 0.9166666666666666
```